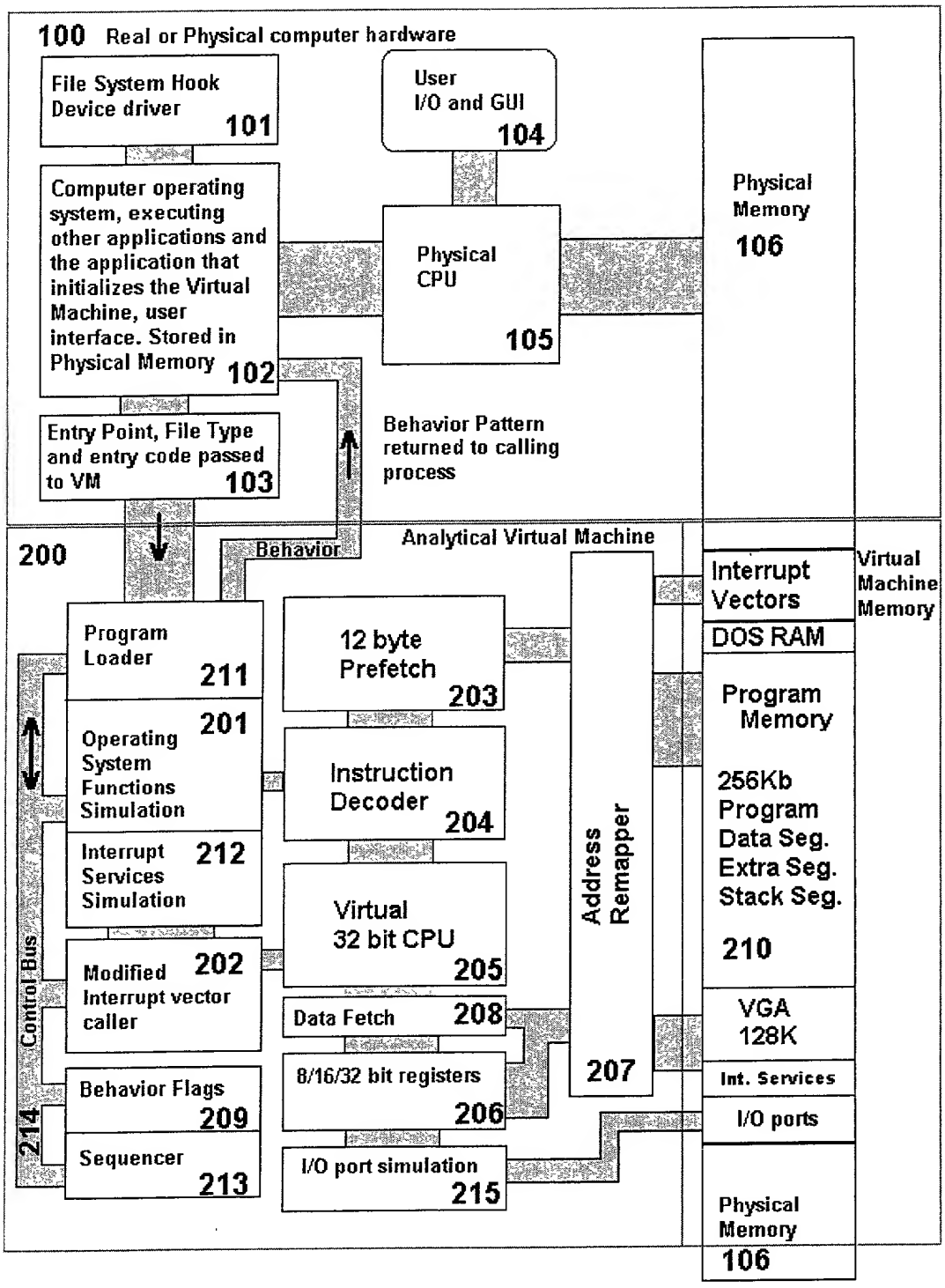

Figure 1: The vCIS Analytical Virtual Machine

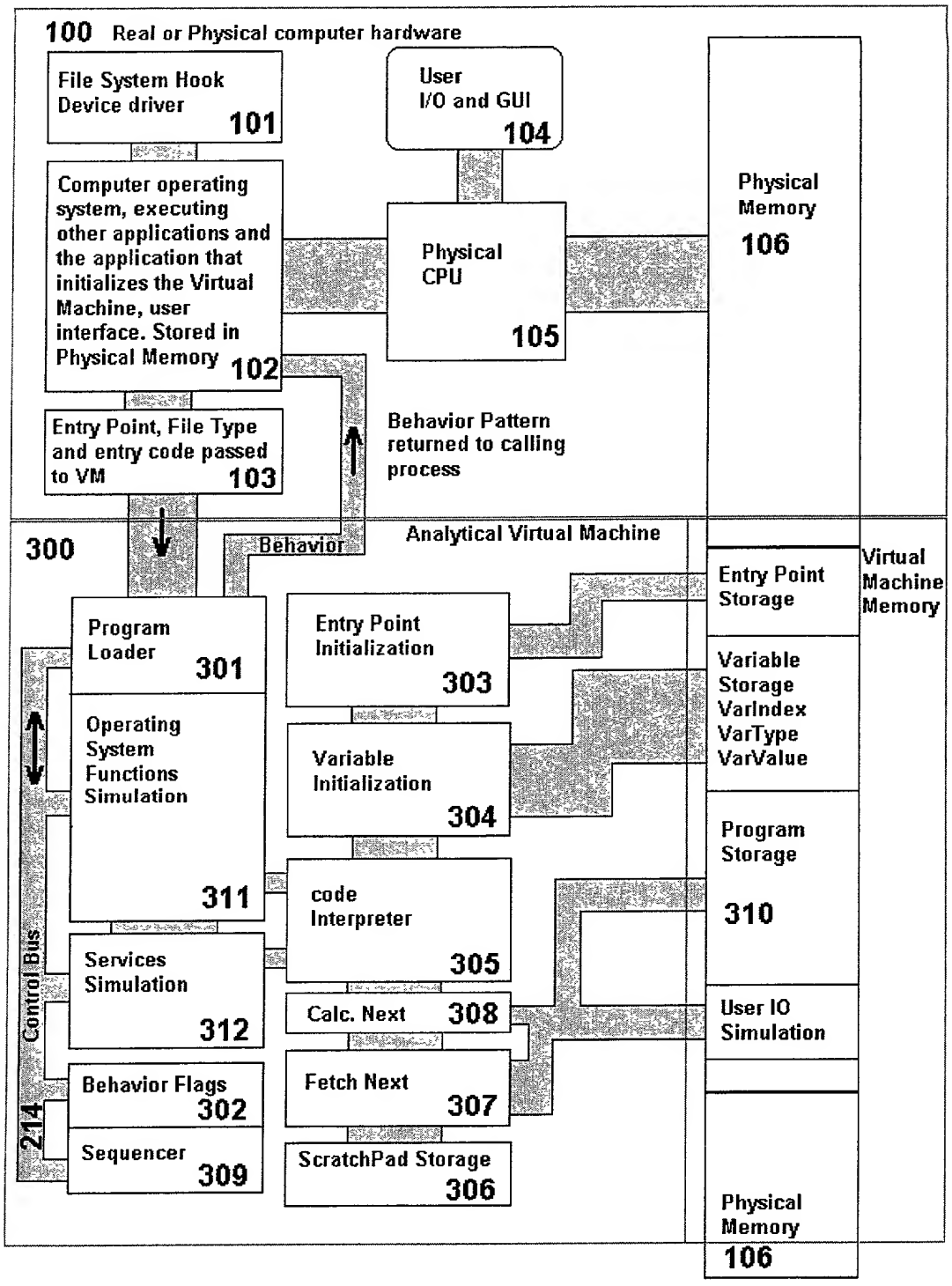

Figure 2: The vCIS VM for execution of HLL program code

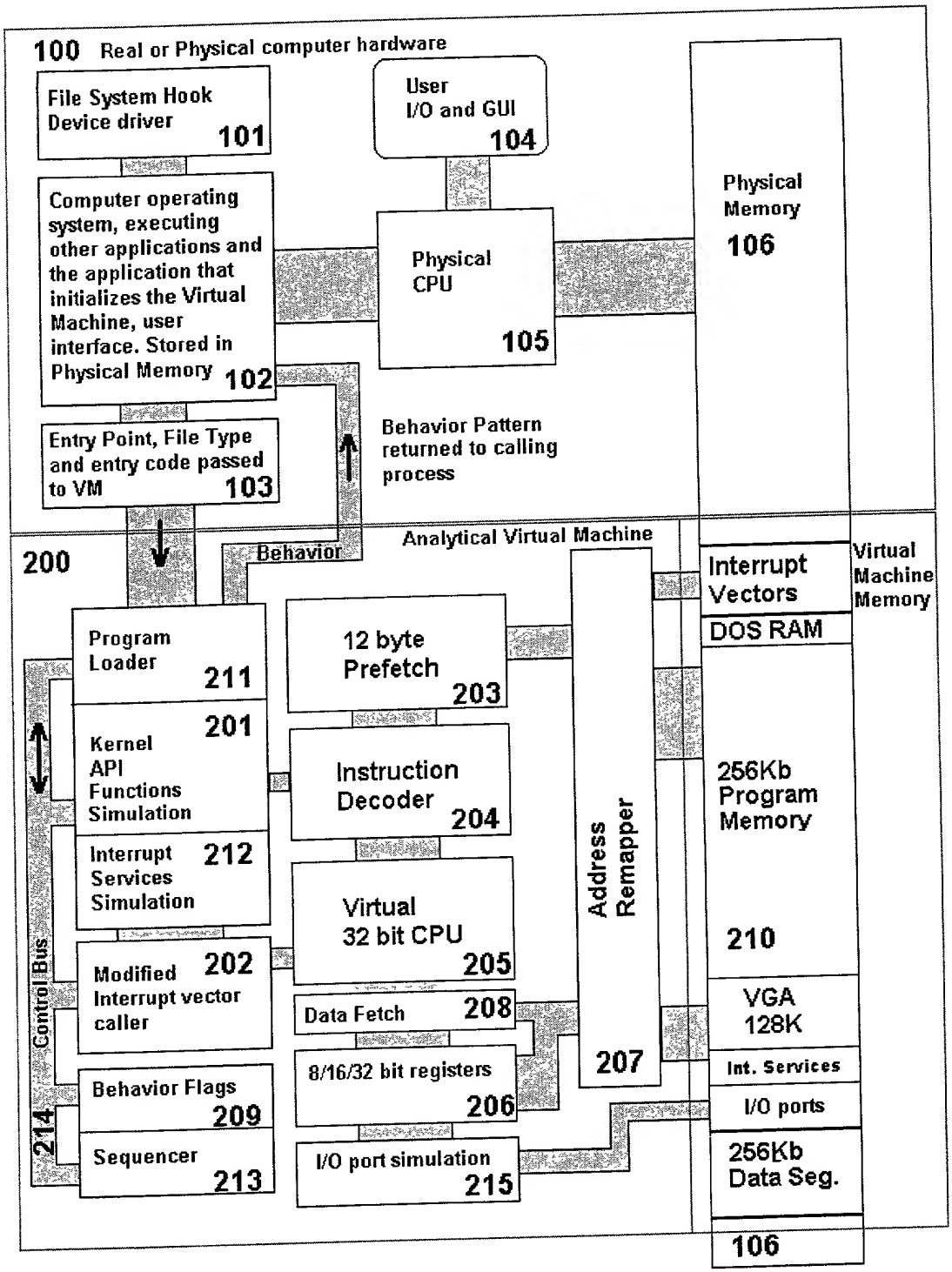

Figure 3: The vCIS Analytical Virtual Machine running 32bit code

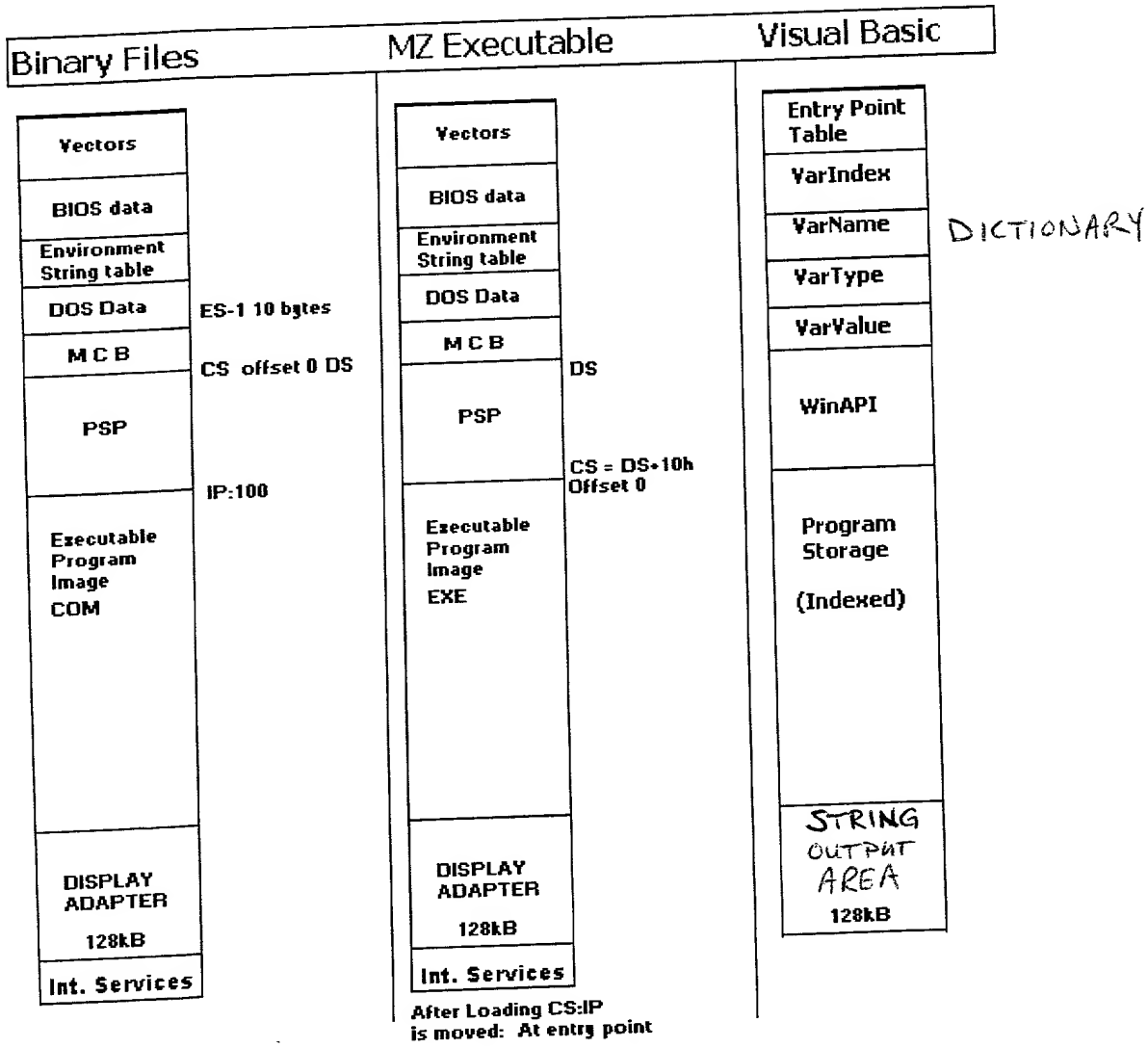

Figure 4: Analytical Virtual Machine memory maps for 3 different operating modes.

Bit	Label	Description
0	OldEntry	Program code contains the previous recorded entry point code and offset
1	Version	Version information in the file is unchanged
2	Encryption	Code contains a decryption (self modifying) procedure
3	SelfMod	Code modifies its own functionality
4	InterruptMod	Code modifies the Interrupt Vector Table contents
5	Jump	Relative Jump near entry point of the code
6	Tunnel	Contains Interrupt tunneling through Int1 or Int3 Trap flag
7	Attach	Contains a procedure that copies this code to the end of other executables
8	ExeSize	Gets the size of an executable
9	ExeAccess	Opens an executable file
10	ExeWrite	Contains code that writes to an executable
11	ExeRead	Contains code that reads code from an executable
12	ExeSearch	Contains a search procedure that looks for executables in this directory
13	ExeSearchRpt	The search procedure is repeated
14	ExeKill	The code kills executables or source files
15	DirKill	The code kills entire directories
16	Reloc	Code relocates itself in memory
17	MemAlloc	Code allocates memory blocks to itself
18	MemStealth	Code labels memory control blocks (MCB's) as owned by operating system
19	FlexEntry	Code is relocatable
20	DirectAccess	Code attempts to directly access the hard disk drive (HDD)
21	TSR	Code terminates but stays resident
22	Chained	Code loads another executable and passes control to it.
23	Ring0	Code contains a call gate to ring0
24	DataOverlap	Code and Data segments overlap, creating a writable code segment
25	ReEntry	Recursive re-entrant code
26	Overwrite	Overwrites files on HDD
27	ExeHdr	File format is inconsistent
28	EmuFail	Code failed to run in virtual environment
29	StandardSys	takes over a standard operating system, or attaches to standard OS function
30	IntRoutineAdd	Code contains a Interrupt Service Routine to which a new IVT entry is mapped
31	HWBios	Flashes the BIOS ROM with non-BIOS code (format of code)
32	EntryOut	Entry to code is not within code segment but in Data segment

FIG. 5A

Bit	Label	Description
33	MoveEnd	Entry point is near the end of the code segment
34	VSafeOff	A call is mode to the OS to switch off vSafe (a DOS based behavior blocker)
35	WriteDirect	The code attempts to write direct to the HDD hardware
36	MBRinfect	The code attempts to write direct to sector 1, track 0, Head 0 of the HDD
37	SectorSmash	Smashes sectors on the HDD by writing garbage
38	Stealth	Code contains instructions to hide its code from other programs (OS hooking)
39	TimeTrigger	Code contains a function that checks system time and branches accordingly
40	Formats	Calls Format function or API
41	SneakyInt	Calls an interrupt as a far call rather than using INT nn call
42	ReadChksum	Reads checksum value from executable file header
43	WriteChksum	Writes new checksum value to executable file header
44	EntryMod	Changes the Entry point value in the header
45	EntryCodeMod	Writes to the entry point location of an executable file
46	HwintCtl	Writes direct to the Interrupt Controller
47	API	Modifies a system API
48	SectorSize	Sets sector size of NE/PE/LE files to maximum and copies own code there

FIG. 5B

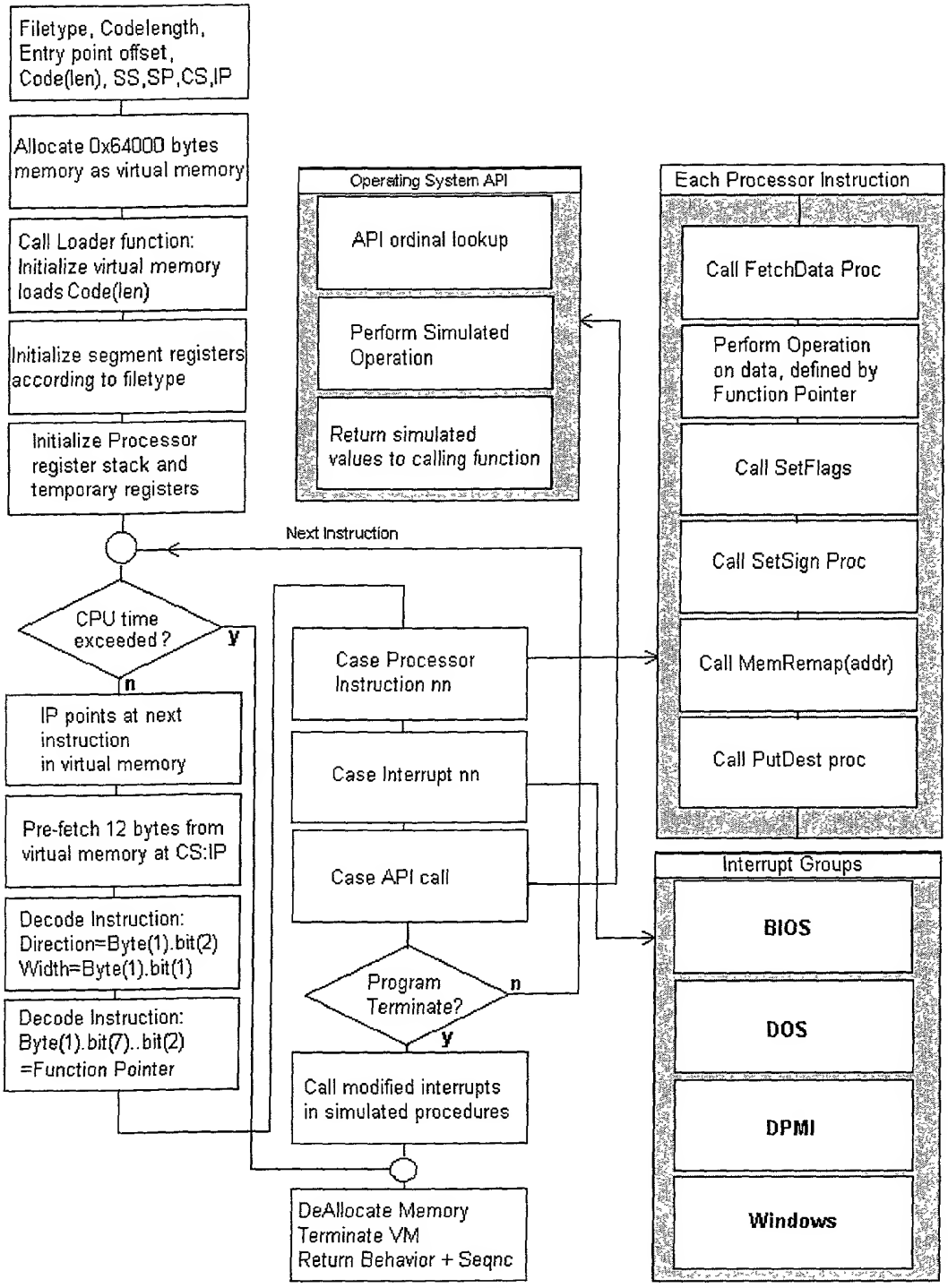

Figure 6: Processing flow within the AVM

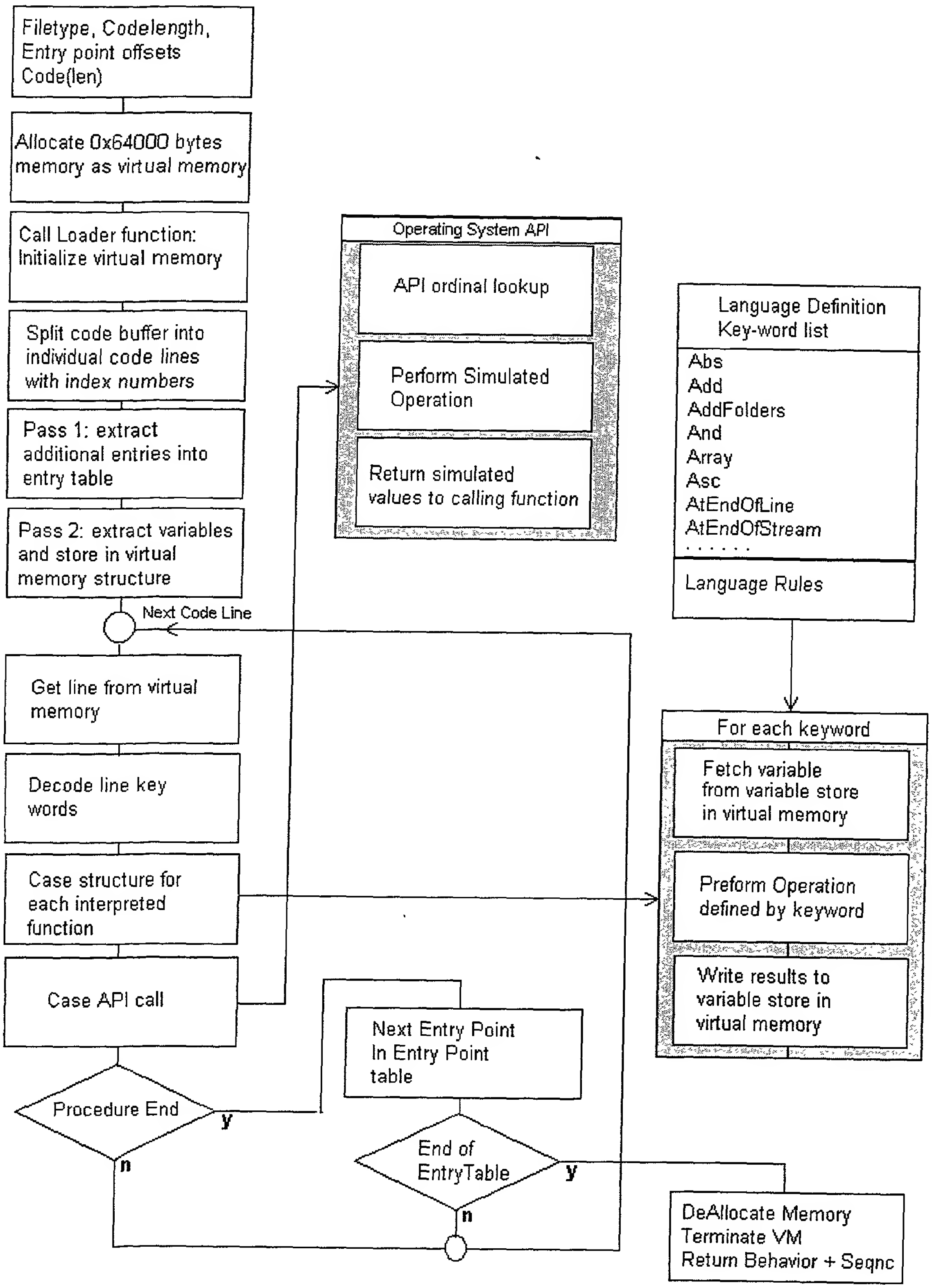
Filetype, Codelength, Entry point offsets Code(len)
Allocate 0x64000 bytes memory as virtual memory
Call Loader function: Initialize virtual memory
Split code buffer into individual code lines with index numbers
Pass 1: extract additional entries into entry table
Pass 2: extract variables and store in virtual memory structure
Next Code Line
Get line from virtual memory
Decode line key words
Case structure for each interpreted function
Case API call
Procedure End
y
n
Operating System API
API ordinal lookup
Perform Simulated Operation
Return simulated values to calling function
Next Entry Point In Entry Point table
End of EntryTable
y
n
Language Definition Key-word list
Abs
Add
AddFolders
And
Array
Asc
AtEndOfLine
AtEndOfStream
. . . . . .
Language Rules
For each keyword
Fetch variable from variable store in virtual memory
Preform Operation defined by keyword
Write results to variable store in virtual memory
DeAllocate Memory Terminate VM Return Behavior + Seqnc

FIG. 7